**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP**

**KHOA ĐIỆN TỬ**

**Bộ môn: Công nghệ thông tin**

****

**BÀI TẬP KẾT THÚC MÔN HỌC**

MÔN HỌC

**LẬP TRÌNH PYTHON**

Sinh viên : Hoàng Kim Ngọc

Lớp : K58KTP.K01

Giáo viên giảng dạy: TS. Nguyễn Văn Huy

Link Github: https://github.com/HoangKimNgoc123/PYTHON\_exam-.gitâccaaaascacacscaccahttps://github.com/HoangKimNgoc123/PYTHON\_exam-.githttps://github.com/HoangKimNgoc123/PYTHON\_exam-.git

https://github.com/HoangKimNgoc123/PYTHON\_exam-.git

**THÁI NGUYÊN – 2025**

|  |  |
| --- | --- |
| **TRƯỜNG ĐHKTCN** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM** |
| **KHOA ĐIỆN TỬ** | ***Độc lập - Tự do - Hạnh phúc*** |

**BÀI TẬP KẾT THÚC MÔN HỌC**

**Môn học: Lập trình Python**

**Bộ môn: Công nghệ thông tin**

**Sinh viên :**  Hoàng Kim Ngọc **MSSV:** K225480106053

**Lớp :**  K58KTP.K01      **Ngành:** Kỹ thuật máy tính

**Giáo viên hướng dẫn:** TS. Nguyễn Văn Huy

**Ngày giao đề:** 20/05/2025 **Ngày hoàn thành:** 10/06/2025

**Tên đề tài :** Critter Caretaker với GUI

**Yêu cầu  :** Xây ứng dụng Critter Caretaker cho phép tạo, cho ăn, cho ngủ critter qua giao diện.

|  |
| --- |
| GIÁO VIÊN HƯỚNG DẪN |
| *(Ký và ghi rõ họ tên)* |

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

................................................................................................................................................................................................................................................................................................................................

Thái Nguyên, ngày….tháng…..năm 2025

               Giáo viên hướng dẫn

*(Ký ghi rõ họ tên)*

**MỤC LỤC**

[LỜI CAM ĐOAN 5](#_Toc3279)

[DANH MỤC CÁC BẢNG VÀ HÌNH VẼ, ĐỒ THỊ 6](#_Toc25334)

[LỜI NÓI ĐẦU 7](#_Toc9655)

[CHƯƠNG 1: GIỚI THIỆU ĐỀ BÀI 1](#_Toc18694)

[1.1 Đề bài 1](#_Toc10966)

[1.2 Phân tích đề bài 1](#_Toc14045)

[1.3 Mục tiêu và hướng giải quyết 1](#_Toc80)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 3](#_Toc5517)

[2.1 Lập trình hướng đối tượng (OOP) 3](#_Toc4741)

[2.2 Thư viện Tkinter 4](#_Toc28993)

[CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH 6](#_Toc28951)

[3.1 Sơ đồ khối hệ thống 6](#_Toc1706)

*[3.1.1 Các module trong chương trình](#_Toc2844)* [6](#_Toc2844)

*[3.1.2 Sơ đồ phân cấp chức năng](#_Toc24816)* [6](#_Toc24816)

[3.2 Sơ đồ khối các thuật toán chính 6](#_Toc24041)

*[3.2.1. Sơ đồ khối thuật toán Tạo Critter](#_Toc13264)* [6](#_Toc13264)

*[3.2.2. Sơ đồ khối thuật toán Cho ăn](#_Toc13075)* [7](#_Toc13075)

*[3.2.3. Sơ đồ khối thuật toán chơi](#_Toc15382)* [8](#_Toc15382)

*[3.2.4. Sơ đồ khối thuật toán Ngủ](#_Toc12184)* [9](#_Toc12184)

[3.3 Cấu trúc dữ liệu 10](#_Toc157)

*[3.3.1. Lớp Critter](#_Toc17755)* [10](#_Toc17755)

*[3.3.2. Các biến liên quan trong lớp giao diện (GUI)](#_Toc24684)* [11](#_Toc24684)

*[3.3.3. Dữ liệu tạm thời (local)](#_Toc24905)* [11](#_Toc24905)

[3.4 Chương trình 12](#_Toc32434)

[CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT LUẬN 14](#_Toc11115)

[4.1 Thực nghiệm 14](#_Toc2122)

*[4.1.1 Môi trường thực nghiệm](#_Toc14346)* [14](#_Toc14346)

*[4.1.2 Code và chạy chương trình](#_Toc31207)* [15](#_Toc31207)

*[4.1.3 Đánh giá chương trình](#_Toc5220)* [20](#_Toc5220)

[4.2 Kết luận 21](#_Toc27230)

[TÀI LIỆU THAM KHẢO 24](#_Toc26386)

# LỜI CAM ĐOAN

Em xin cam đoan rằng đề bài “Critter Caretaker với GUI” là kết quả của quá trình nghiên cứu và thực hiện của nhóm chúng em dưới sự hướng dẫn của thầy Nguyễn Văn Huy.

Toàn bộ nội dung trong đề tài này đều do em tự thực hiện, dựa trên các kiến thức đã học và các nguồn tài liệu tham khảo được liệt kê trong danh mục tài liệu tham khảo. Em không sao chép nội dung của bất kỳ cá nhân hoặc tổ chức nào một cách trái phép. Nếu phát hiện có bất kỳ hành vi gian lận hoặc vi phạm quy định về nghiên cứu khoa học, em xin hoàn toàn chịu trách nhiệm trước nhà trường và pháp luật.

Em cam kết mọi thông tin, số liệu và kết quả trong đề tài là chính xác và trung thực.

Thái Nguyên, ngày 10 tháng 06 năm 2025

Sinh viên thực hiện

*(Ký ghi rõ họ tên)*

# DANH MỤC CÁC BẢNG VÀ HÌNH VẼ, ĐỒ THỊ

Hình 1. Mô tả các thành phần chính trong OOP........................................... 3

Hình 2. Sơ đồ phân cấp chức năng của chương trình Critter Caretaker....... 6

Hình 3. Sơ đồ khối thuật toán Tạo Critter ....................................................7

Hình 4. Sơ đồ khối thuật toán Cho ăn...........................................................8

Hình 5. Sơ đồ khối thuật toán Chơi...............................................................9

Hình 6. Sơ đồ khối thuật toán Ngủ...............................................................10

Hình 7. Phần mềm Anaconda.......................................................................14

Hình 8. Phần mềm VScode..........................................................................14

Hình 9. Quá trình code chương trình............................................................15

Hình 10. Giao diện chương trình Critter Caretaker......................................16

Hình 11. Bước nhập tên thú cưng.................................................................16

Hình 12. Giao diện khi nhấn nút “Tạo”........................................................17

Hình 13. Giao diện khi nhấn nút hoạt động đầu tiên “Cho ăn”....................17

Hình 14. Giao diện khi nhấn nút “Chơi”......................................................18

Hình 15. Giao diện khi nhấn nút “Ngủ”.......................................................19

Hình 16. Giao diện khi tạo mới thú cưng.....................................................19

Hình 17. Thông báo lỗi chưa nhập tên thú cưng..........................................20

Hình 18. Thông báo lỗi chưa tạo thú cưng...................................................20

Bảng 1. Bảng cấu trúc dữ liệu trong lớp Critter...........................................11

Bảng 2. Bảng cấu trúc dữ liệu trong lớp giao diện (GUI)............................11

Bảng 3. Bảng cấu trúc dữ liệu local..............................................................12

# LỜI NÓI ĐẦU

Trong quá trình học tập và rèn luyện kỹ năng lập trình, việc áp dụng lý thuyết vào các bài tập thực hành có ý nghĩa vô cùng quan trọng. Môn học Lập trình Python không chỉ cung cấp kiến thức nền tảng về cú pháp và cấu trúc lập trình, mà còn mở ra cơ hội để sinh viên làm quen với tư duy hướng đối tượng, xây dựng giao diện người dùng và giải quyết các bài toán mô phỏng trong thực tế.

Đề tài " Critter Caretaker với GUI " được lựa chọn với mục tiêu vận dụng tổng hợp các kiến thức đã học để xây dựng một chương trình hoàn chỉnh, có tính tương tác cao. Thông qua đề tài này, người thực hiện đã có cơ hội thực hành thiết kế lớp đối tượng, xử lý sự kiện, cập nhật trạng thái động và thiết kế giao diện người dùng bằng thư viện Tkinter.

Mặc dù chỉ là một ứng dụng đơn giản, nhưng quá trình phát triển chương trình đã giúp người thực hiện hiểu rõ hơn về quy trình thiết kế phần mềm, xử lý logic tương tác, cũng như rèn luyện kỹ năng tư duy lập trình hiệu quả.

Báo cáo này trình bày toàn bộ quá trình phân tích, thiết kế, cài đặt và kiểm thử ứng dụng Critter Caretaker. Rất mong nhận được sự góp ý và đánh giá từ thầy cô để đề tài được hoàn thiện tốt hơn. Em xin chân thành cảm ơn thầy Nguyễn Văn Huy và thầy Phạm Sỹ Quang đã tận tình chỉ bảo và hỗ trợ trong suốt quá trình thực hiện đề tài này.

**CHƯƠNG 1: GIỚI THIỆU ĐỀ BÀI**

* 1. **Đề bài**

“Xây ứng dụng Critter Caretaker (Chapter 8) cho phép tạo, cho ăn, cho ngủ critter qua giao diện.

- Yêu cầu đầu vào – đầu ra:

+ Đầu vào: Tên critter từ Entry, các nút hành động.

+ Đầu ra: Trạng thái Critter (hunger, boredom) hiển thị trên Label.

- Tính năng yêu cầu:

+ Class Critter với attributes và methods.

+ GUI: Entry tạo critter, Buttons: “Tạo”, “Cho ăn”, “Chơi”, “Ngủ”.

+ Cập nhật trạng thái real-time.

+ Bắt lỗi khi chưa tạo critter. ”

* 1. **Phân tích đề bài**

Ứng dụng Critter Caretaker là một ứng dụng mô phỏng quá trình chăm sóc một thú cưng ảo (critter) thông qua các thao tác như cho ăn, cho chơi, cho ngủ. Trạng thái của critter sẽ thay đổi theo thời gian và phản ánh qua các chỉ số đói (hunger) và buồn chán (boredom), được cập nhật liên tục trên giao diện. Người dùng phải cho ăn và chơi cùng để giữ cho thú cưng của mình có tâm trạng tốt.

Người dùng cần phải tạo thú cưng bằng cách nhập tên, nếu chưa tạo tên mà thực hiện các tính năng khác thì sẽ báo lỗi. Đối với các tính năng khác như cho ăn, chơi, ngủ thì mỗi lần thú cưng hoạt động, trạng thái của nó được cập nhật theo thời gian thực (real-time) và hiển thị trên giao diện.

**1.3 Mục tiêu và hướng giải quyết**

Đối với đề tài này, ứng dụng cần được xây dựng theo các mục tiêu như sau:

- Thiết kế giao diện trực quan và thân thiện với người dùng.

- Xử lý các trạng thái động của critter và cập nhật chúng hợp lý theo hành vi người dùng.

- Đảm bảo chương trình chạy ổn định, không xảy ra lỗi khi người dùng thao tác chưa đúng trình tự.

Để giải quyết các vấn đề trên, đề tài đã vận dụng tổng hợp các kiến thức sau:

- Lập trình hướng đối tượng (*class,* *object*, *method*).

- Quản lý trạng thái và logic trong thời gian thực.

- Thiết kế giao diện GUI bằng Tkinter, xử lý sự kiện người dùng và cập nhật nội dung động.

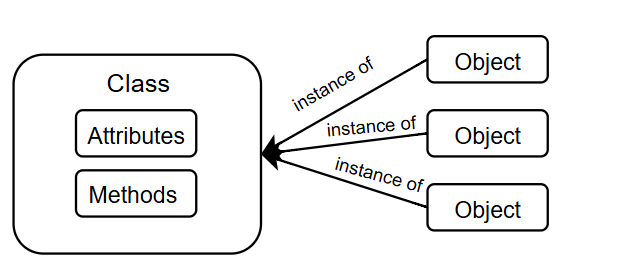
*Tóm tắt chương 1*

*Chương này trình bày khái quát đề tài, nêu bật yêu cầu xây dựng một ứng dụng Python có giao diện đồ họa (GUI) cho phép người dùng tạo và chăm sóc một thú cưng ảo. Đồng thời, chương cũng phân tích các tính năng chính, khó khăn gặp phải và định hướng vận dụng kiến thức lập trình hướng đối tượng, GUI với Tkinter và xử lý tương tác.*

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

**2.1 Lập trình hướng đối tượng (OOP)**

Lập trình hướng đối tượng ([tiếng Anh](https://vi.wikipedia.org/wiki/Ti%E1%BA%BFng_Anh" \o "Tiếng Anh): object-oriented programming - OOP) là một [mẫu hình lập trình](https://vi.wikipedia.org/wiki/M%E1%BA%ABu_h%C3%ACnh_l%E1%BA%ADp_tr%C3%ACnh" \o "Mẫu hình lập trình) dựa trên khái niệm "[đối tượng](https://vi.wikipedia.org/wiki/%C4%90%E1%BB%91i_t%C6%B0%E1%BB%A3ng_(khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh)" \o "Đối tượng (khoa học máy tính))" (object), mà trong đó, đối tượng chứa đựng các [dữ liệu](https://vi.wikipedia.org/wiki/D%E1%BB%AF_li%E1%BB%87u" \o "Dữ liệu) trong các [trường](https://vi.wikipedia.org/wiki/Tr%C6%B0%E1%BB%9Dng_(khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh)" \o "Trường (khoa học máy tính)), thường được gọi là các *thuộc tính* (attributes); và mã nguồn, được tổ chức thành các *[phương thức](https://vi.wikipedia.org/wiki/Ph%C6%B0%C6%A1ng_th%E1%BB%A9c_(l%E1%BA%ADp_tr%C3%ACnh_m%C3%A1y_t%C3%ADnh)" \o "Phương thức (lập trình máy tính))* (methods).



*Hình 1. Mô tả các thành phần chính trong OOP*

Tính chất cơ bản của lập trình hướng đối tượng bao gồm:

- Tính đóng gói (Encapsulation): Encapsulation là một kỹ thuật lập trình nền tảng được sử dụng để gom nhóm các attributes và methods cần thiết vào một Object. Kỹ thuật này còn được biết với tên gọi khác là Data/Information Hiding.

Data Hiding trong lập trình hướng đối tượng OOP thường dùng để bảo vệ các thành phần bên trong của Object (thường gọi là private). Các thành phần bên ngoài sẽ không được can thiệp và sử dụng các thành phần này trực tiếp mà phải thông qua các phương thức công khai (public).

- Tính trừu tượng (Abstraction): Abstraction trong lập trình hướng đối tượng (OOP) là một kỹ thuật cho phép người lập trình tập trung vào những gì một đối tượng làm mà không cần phải quan tâm đến cách nó thực hiện. Điều này được thực hiện bằng cách định nghĩa các phương thức ở dạng mô tả chung mà không cần chi tiết cụ thể. Tính trừu tượng giúp giảm sự phức tạp của chương trình và tăng cường khả năng tái sử dụng mã nguồn.

- Tính kế thừa (Inheritance): Inheritance trong lập trình hướng đối tượng OOP là một cơ chế xây dựng class mới dựa trên các class đã có. Các class kế thừa sẽ bao gồm toàn bộ các attributes và methods từ base class (lớp cơ sở) hay parent class (lớp cha). Inheritance trong OOP bao gồm 2 loại chính:

+ Single Inheritance (đơn kế thừa): class chỉ có thể kế thừa từ một parent class duy nhất.

+ Multiple inheritance (đa kế thừa): class có thể kế thừa từ nhiều parent class.

- Tính đa hình (Polymorphism): Polymorphism trong lập trình hướng đối tượng OOP cho phép một Object có thể có nhiều hình dạng và hành vi khác nhau. Polymorphism trong OOP được chia làm 2 loại:

+ Static Polymorphism (đa hình tĩnh): là cơ chế định nghĩa lại các methods cùng tên, nhưng có thể khác số lượng hoặc kiểu của tham số. Static Polymorphism còn được gọi là Method Overloading.

+ Dynamic Polymorphism (đa hình động): là cơ chế định nghĩa lại các methods cùng tên, cùng tham số và kiểu trả về từ parent class. Dynamic Polymorphism còn được gọi là Method Overriding.

Sự khác biệt lớn nhất giữa Static và Dynamic Polymorphism là: Static Polymorphism được xử lý tại thời điểm biên dịch (compile-time). Dynamic Polymorphism được xử lý tại thời điểm chạy chương trình (runtime).

**2.2 Thư viện Tkinter**

Tkinter là một thư viện trong ngôn ngữ lập trình Python được sử dụng để tạo giao diện đồ họa người dùng (GUI). "Tkinter" là viết tắt của "Tk interface,", một toolkit đồ họa cung cấp các công cụ để phát triển giao diện người dùng.

Tkinter là một phần của thư viện tiêu chuẩn của Python và đã được tích hợp sẵn trong hầu hết các cài đặt Python. Điều này giúp cho Tkinter trở thành một lựa chọn phổ biến cho việc phát triển ứng dụng với giao diện đồ họa đơn giản trong Python.

Một số đặc điểm của Tkinter bao gồm khả năng tạo các thành phần giao diện như cửa sổ, nút, ô văn bản, và các widget khác để tương tác với người dùng. Tkinter cung cấp cả các sự kiện và phương thức để xử lý tương tác người dùng và thay đổi trạng thái của ứng dụng.

*Tóm tắt chương 2*

*Chương này trình bày hai nền tảng kỹ thuật chính được sử dụng trong đề tài: lập trình hướng đối tượng (OOP) và thư viện Tkinter của Python. OOP giúp xây dựng mô hình thú cưng ảo một cách rõ ràng, thông qua việc định nghĩa lớp Critter với các thuộc tính và hành vi cụ thể. Thư viện Tkinter được sử dụng để xây dựng giao diện người dùng đồ họa (GUI), cho phép người dùng tương tác trực tiếp với chương trình thông qua các thành phần như ô nhập liệu, nút bấm, nhãn hiển thị và hộp thoại thông báo.*

**CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH**

**3.1 Sơ đồ khối hệ thống**

*3.1.1 Các module trong chương trình*

Chương trình được chia thành hai module chính:

- Module logic *(Critter Class)*: Quản lý toàn bộ trạng thái và hành vi của critter (đói, buồn chán, ngủ…).

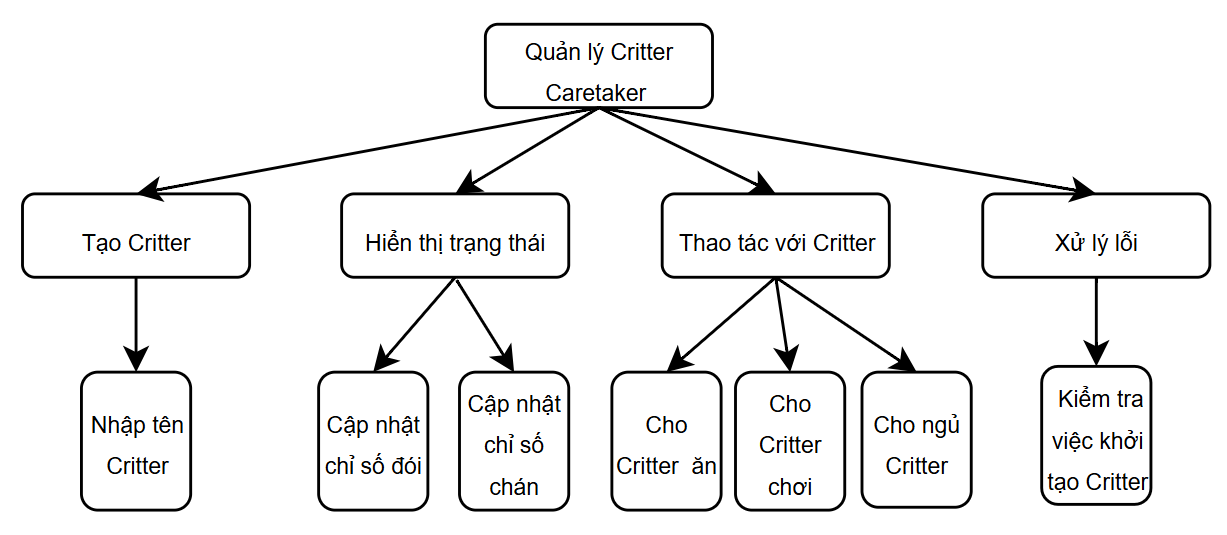
- Module giao diện *(CritterApp Class)*: Tạo cửa sổ GUI, bố trí các thành phần như Entry, Label, Button và xử lý tương tác người dùng với logic.

Ngoài ra, còn có một số thành phần bổ trợ:

- Module sự kiện: Liên kết các nút trên giao diện với hành vi thực tế của Critter.

- Module thông báo lỗi: Hiển thị thông báo nếu người dùng thao tác sai quy trình (chưa tạo critter...).

*3.1.2 Sơ đồ phân cấp chức năng*



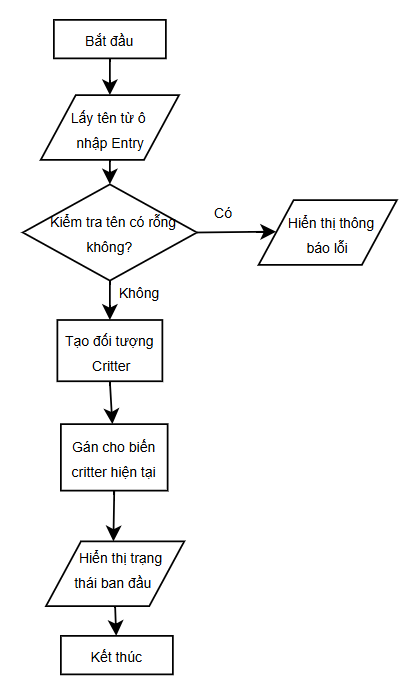
*Hình 2. Sơ đồ phân cấp chức năng của chương trình Critter Caretaker*

**3.2 Sơ đồ khối các thuật toán chính**

*3.2.1. Sơ đồ khối thuật toán Tạo Critter*

Chức năng: Khởi tạo một đối tượng Critter mới từ tên nhập vào.

Luồng xử lý:

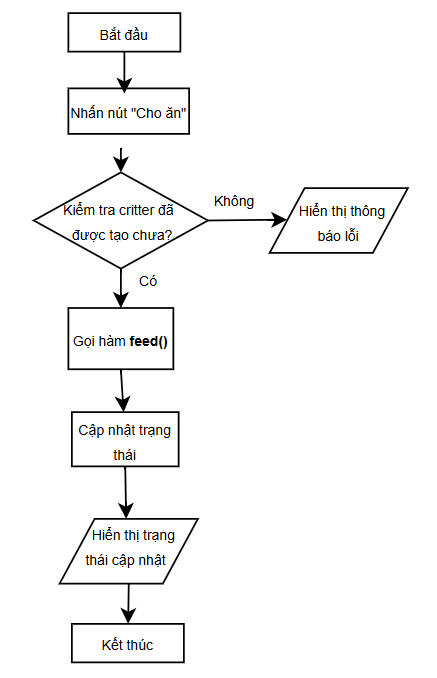


*Hình 3. Sơ đồ khối thuật toán Tạo Critter*

*3.2.2. Sơ đồ khối thuật toán Cho ăn*

Chức năng: Giảm chỉ số hunger của critter khi người dùng nhấn nút "Cho ăn".

Luồng xử lý:

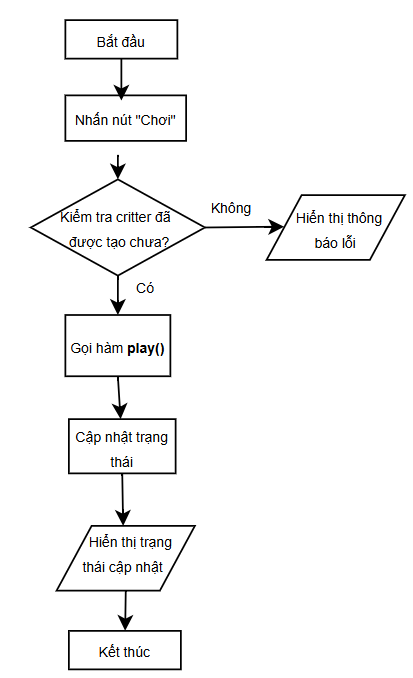


Hình 4. Sơ đồ khối thuật toán Cho ăn

*3.2.3. Sơ đồ khối thuật toán chơi*

Chức năng: Giảm chỉ số boredom của critter khi người dùng nhấn nút "Chơi".

Luồng xử lý:

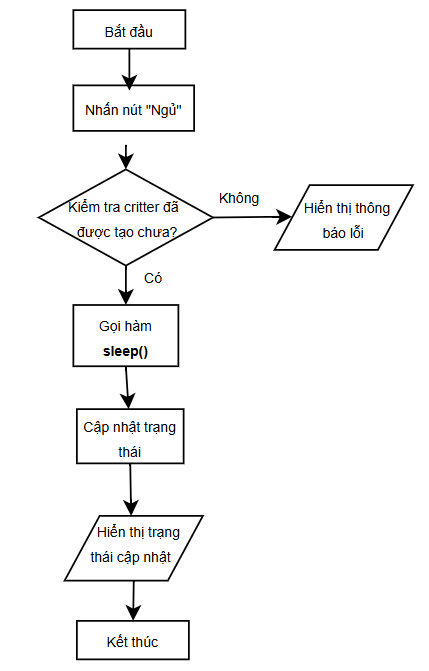


Hình 5. Sơ đồ khối thuật toán Chơi

*3.2.4. Sơ đồ khối thuật toán Ngủ*

Chức năng: Thay đổi trạng thái critter khi người dùng nhấn "Ngủ" (giảm nhẹ cả hunger và boredom).

Luồng xử lý:



*Hình 6. Sơ đồ khối thuật toán Ngủ*

**3.3 Cấu trúc dữ liệu**

Chương trình Critter Caretaker sử dụng mô hình lập trình hướng đối tượng (OOP), do đó cấu trúc dữ liệu chủ yếu tập trung vào lớp (class) Critter, và các biến đối tượng đi kèm.

*3.3.1. Lớp Critter*

Đây là lớp trung tâm, đại diện cho mỗi thú cưng (critter) mà người dùng tạo ra. Mỗi đối tượng thuộc lớp này sẽ lưu trữ và quản lý thông tin trạng thái của một cá thể critter.

|  |  |  |
| --- | --- | --- |
| **Thuộc tính (Attribute)** | **Kiểu dữ liệu** | **Ý nghĩa** |
| name | str | Tên của critter do người dùng nhập vào |
| hunger | int | Mức độ đói của critter (càng cao càng đói) |
| boredom | int | Mức độ buồn chán của critter (càng cao càng chán) |

*Bảng 1. Bảng cấu trúc dữ liệu trong lớp Critter*

*3.3.2. Các biến liên quan trong lớp giao diện (GUI)*

Trong lớp CritterApp (lớp điều khiển GUI), các cấu trúc dữ liệu chính bao gồm:

|  |  |  |
| --- | --- | --- |
| **Tên biến** | **Kiểu dữ liệu** | **Ý nghĩa** |
| self.critter | Critter hoặc None | Biến lưu đối tượng critter hiện tại |
| self.name\_entry | tk.Entry | Ô nhập tên critter từ người dùng |
| self.status\_label | tk.Label | Nhãn hiển thị trạng thái hiện tại của critter |
| self.feed\_button, self.play\_button, self.sleep\_button | tk.Button | Các nút điều khiển hành vi của critter |

*Bảng 2. Bảng cấu trúc dữ liệu trong lớp giao diện (GUI)*

*3.3.3. Dữ liệu tạm thời (local)*

Ngoài các thuộc tính đối tượng, chương trình còn có các biến tạm dùng trong các phương thức:

|  |  |  |
| --- | --- | --- |
| **Tên biến** | **Kiểu dữ liệu** | **Ý nghĩa** |
| name | str | Dữ liệu nhập từ Entry để khởi tạo Critter |
| status | str | Chuỗi mô tả trạng thái của critter (hiển thị trên giao diện) |

*Bảng 3. Bảng cấu trúc dữ liệu local*

**3.4 Chương trình**

Lớp Critter là thành phần trung tâm xử lý logic của thú cưng ảo. Nó có các thuộc tính đặc trưng là *name*, *hunger* (mức độ đói), và *boredom* (mức độ buồn chán). Mỗi hành động như ăn, chơi, hay ngủ đều ảnh hưởng đến các trạng thái này. Dưới đây là mô tả chi tiết các phương thức:

***\_\_init\_\_(self, name):***

+ Chức năng: Khởi tạo một đối tượng Critter mới với tên do người dùng nhập. Trạng thái ban đầu là không đói và không buồn chán.

+ Tham số: *name* (chuỗi) – tên của critter.

+ Tác động: Gán *hunger = 0, boredom = 0*.

***pass\_time(self):***

+ Chức năng: Mô phỏng thời gian trôi qua – làm tăng mức độ đói và buồn chán của critter.

+ Tác động: *hunger += 1, boredom += 1.*

+ Đặc biệt: Hàm này được gọi tự động sau mỗi hành động để khiến Critter “già đi” theo thời gian.

***eat(self):***

+ Chức năng: Thực hiện hành động “ăn” – giúp giảm đói.

+ Tác động: Giảm *hunger* đi 2 đơn vị, không xuống dưới 0 (dùng hàm *max(0, ...)*).

+ Sau đó gọi *pass\_time()* để tăng thời gian.

***play(self):***

+ Chức năng: Thực hiện hành động “chơi” – giúp giảm buồn chán.

+ Tác động: Giảm *boredom* đi 2 đơn vị, không xuống dưới 0.

+ Sau đó gọi *pass\_time()* để tăng thời gian.

**sleep(self):**

+ Chức năng: Cho Critter “ngủ” – giúp giảm buồn chán mạnh hơn.

+ Tác động: Giảm *boredom* đi 3 đơn vị, không xuống dưới 0 và tăng *hunger* lên 1 đơn vị (ngủ dậy thì đói).

+ Gọi *pass\_time()* để tiếp tục tăng thời gian.

*Tóm tắt chương 3*

*Chương này đi sâu vào cấu trúc và thiết kế phần mềm. Bao gồm sơ đồ khối hệ thống, phân tích chức năng chính, thuật toán điều khiển trạng thái thú cưng, cấu trúc dữ liệu sử dụng và chi tiết các hàm đã triển khai. Giao diện người dùng được tổ chức trực quan với Entry, Label, Button và logic được tích hợp thông qua các callback tương ứng.*

**CHƯƠNG 4: THỰC NGHIỆM VÀ KẾT LUẬN**

**4.1 Thực nghiệm**

*4.1.1 Môi trường thực nghiệm*

a. Anaconda



*Hình 7. Phần mềm Anaconda*

Anaconda là một bản phân phối Python mạnh mẽ, được thiết kế đặc biệt để hỗ trợ các lĩnh vực như khoa học dữ liệu, phân tích số liệu và phát triển ứng dụng Python. Anaconda tích hợp sẵn nhiều thư viện cần thiết và công cụ quản lý môi trường ảo như:

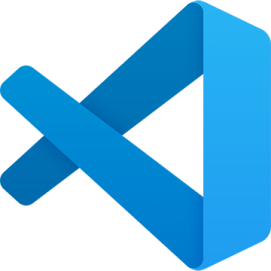
+ Conda – dùng để tạo, quản lý môi trường Python độc lập;

+ Navigator – giao diện đồ họa để khởi động các IDE như JupyterLab, Spyder, VSCode;

+ Hỗ trợ tốt cho các thư viện như *numpy, matplotlib, tkinter, v.v.*

Việc sử dụng Anaconda giúp đảm bảo các thư viện hoạt động ổn định và tránh xung đột môi trường.

b. Visual Studio Code (VSCode)



*Hình 8. Phần mềm VScode*

VSCode là một trình soạn thảo mã nguồn nhẹ, miễn phí do Microsoft phát triển. VSCode hỗ trợ nhiều ngôn ngữ lập trình, đặc biệt có khả năng mở rộng mạnh mẽ thông qua extensions (phần mở rộng).

Trong đề bài này, VSCode được tích hợp trong Anaconda Navigator, giúp tự động kết nối với môi trường Python trong Anaconda. Nhờ đó:

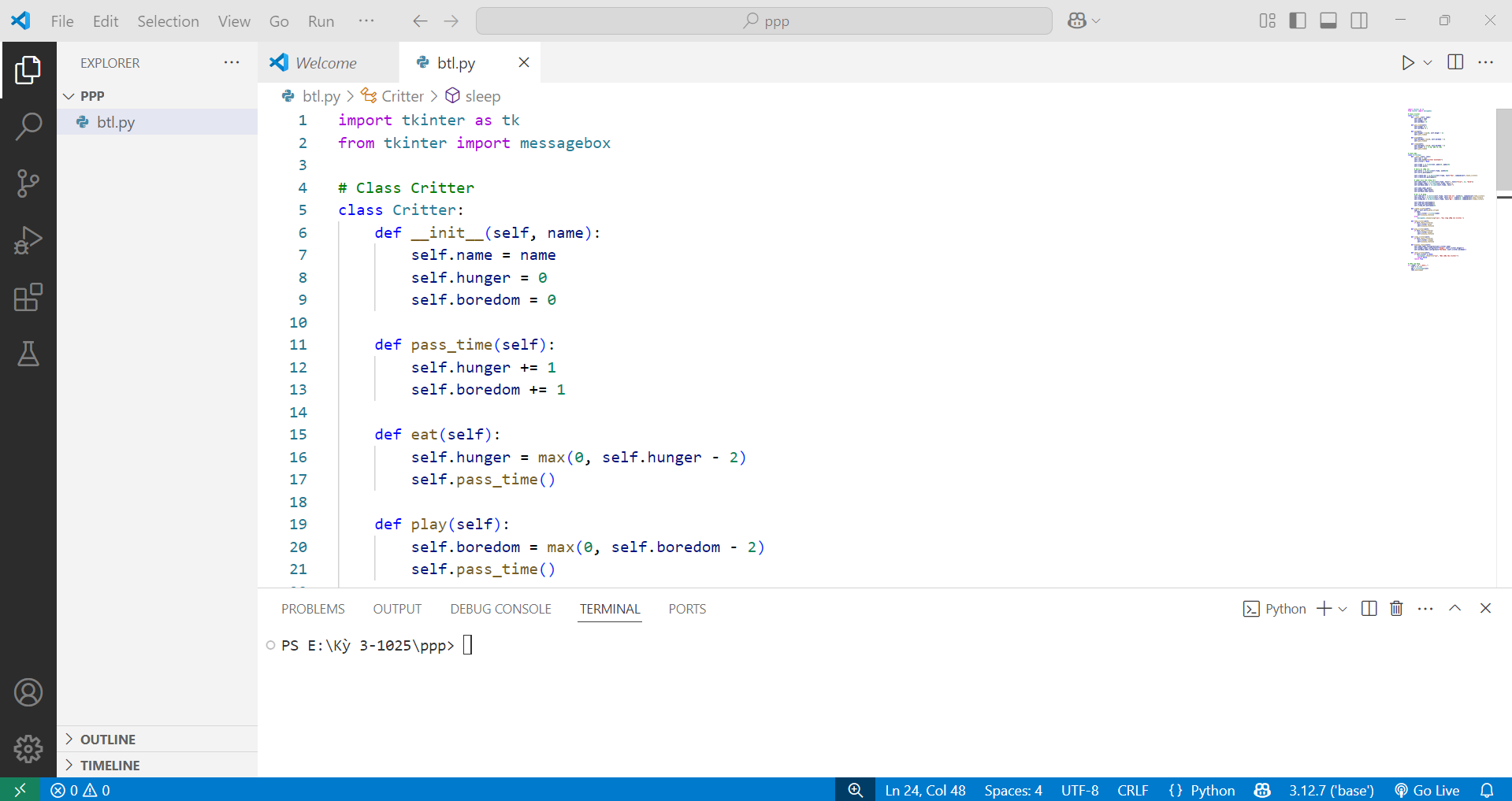
+ Không cần cài đặt lại Python hoặc cấu hình rườm rà;

+ Việc chạy chương trình *tkinter* diễn ra mượt mà, không lỗi;

+ Quản lý môi trường và gói cài đặt đơn giản hơn.

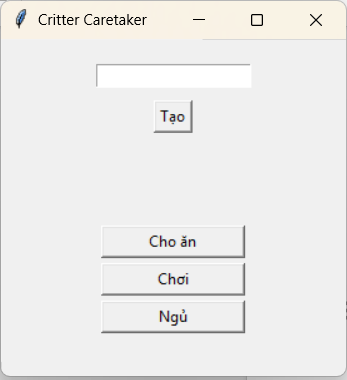
*4.1.2 Code và chạy chương trình*

*-* Tiến hành code chương trình, *import* thư viện *tkinter* và code các lớp đã thiết kế.



*Hình 9. Quá trình code chương trình*

Sau khi hoàn thiện code, trên thanh Terminal, gõ ***python btl.py*** để chạy chương trình.



*Hình 10. Giao diện chương trình Critter Caretaker*

Giao diện chương trình bao gồm:

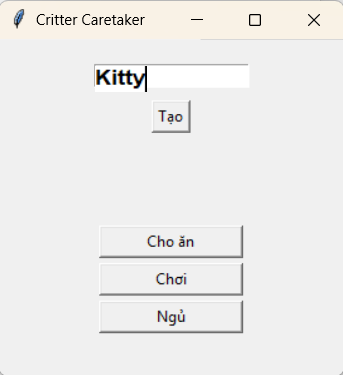
+ Có nút *entry* để nhập tên critter.

+ Có nút “Tạo” để tạo thú cưng.

+ Có 3 nút “Cho ăn”, “Chơi”, “Ngủ”.

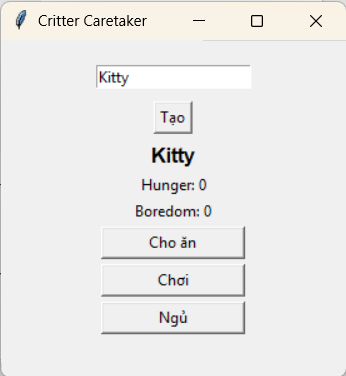
- Tiến hành thử nghiệm chương trình:

+ Nhập tên thú cưng



*Hình 11. Bước nhập tên thú cưng*

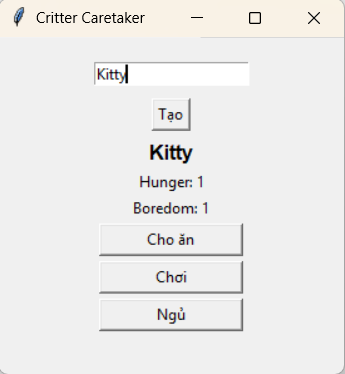
*+* Nhấn nút “Tạo”.



*Hình 12. Giao diện khi nhấn nút “Tạo”*

Sau khi nhấn nút “Tạo”, các trạng thái của thú cưng có tên là “Kitty” được hiển thị, bao gồm *hunger = 0*  và  *boredom = 0*, đây là trạng thái ban đầu khi thú cưng chưa thực hiện hoạt động nào.

+ Nhấn nút hoạt động “Cho ăn”.

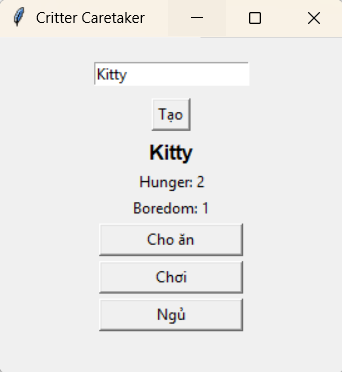


*Hình 13. Giao diện khi nhấn nút hoạt động đầu tiên “Cho ăn”*

Sau khi nhấn nút “Cho ăn”, trạng thái *hunger* sẽ giảm đi 2, tức là

*hunger = 0-2 = -2*, nhưng trạng thái của thú cưng thì không thể âm nên sẽ lấy giá trị là 0 ( sử dụng hàm *max()*), hàm *pass\_time()* cũng được chạy sau đó để mô phỏng thời gian đang chạy khi hoạt động nên trạng thái hunger được cộng thêm 1; trạng thái boredom cũng tương tự.

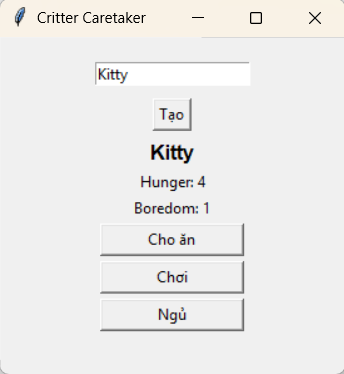
+ Nhấn nút “Chơi”.



Hình 14. Giao diện khi nhấn nút “Chơi”

Khi nhấn nút “Chơi”, trạng thái boredom = 1-2 = -1, nhưng trạng thái của thú cưng thì không thể âm nên sẽ lấy giá trị boredom = 0, hàm pass\_time() được chạy ngay sau đó nên tăng thêm 1, trạng thái hunger không ảnh hưởng nhưng hàm pass\_time() vẫn chạy nên sẽ bị tăng thêm 1, dễ hiểu đơn giản là thời gian trôi qua thì sẽ đói thêm.

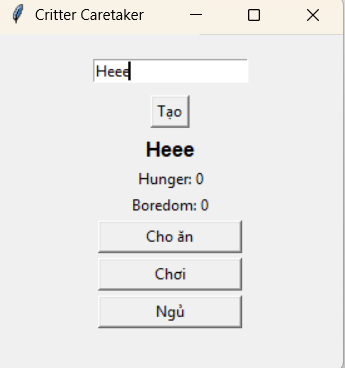
+ Nhấn nút “Ngủ”.



*Hình 15. Giao diện khi nhấn nút “Ngủ”*

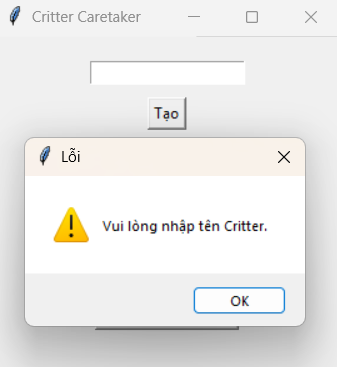
Sau khi nhấn nút ngủ thì trạng thái boredom = 1- 3= -2, tương tự như các trạng thái khác nên ta có giá trị là bordom = 1, trạng thái hunger tăng thêm 2 tính cả sau khi chạy pass\_time().

+ Muốn tạo thú cưng khác thì chỉ cần nhập tên khác rồi nhấn “Tạo”.



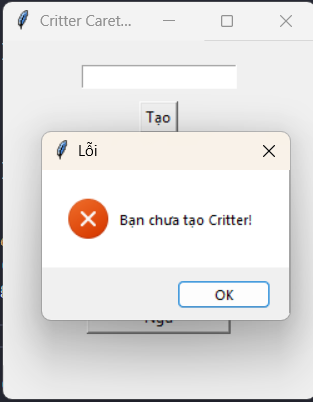
*Hình 16. Giao diện khi tạo mới thú cưng*

+ Trường hợp chưa nhập tên mà đã nhấn tạo thú cưng thì sẽ hiển thị thông báo lỗi.



*Hình 17. Thông báo lỗi chưa nhập tên thú cưng.*

+ Trường hợp chưa tạo thú cưng mà đã thực hiện các hoạt động chơi, cho ăn, ngủ thì cũng sẽ hiển thị lỗi.



*Hình 18. Thông báo lỗi chưa tạo thú cưng*

*4.1.3 Đánh giá chương trình*

- Ưu điểm:

+ Giao diện thân thiện và trực quan.

+ Sử dụng thư viện tkinter giúp xây dựng giao diện đồ họa dễ sử dụng, phù hợp với người dùng phổ thông.

+ Các nút chức năng được bố trí rõ ràng: “Tạo”, “Cho ăn”, “Chơi”, “Ngủ”.

+ Ứng dụng lập trình hướng đối tượng (OOP).

+ Mã nguồn được tổ chức tốt với hai lớp riêng biệt: Critter (xử lý logic) và CritterApp (xử lý giao diện).

+ Dễ bảo trì, mở rộng tính năng như thêm trạng thái “vui vẻ” hoặc “mệt mỏi”.

+ Xử lý lỗi đầy đủ.

+ Có kiểm tra khi nhập tên trống hoặc thao tác với Critter chưa tạo.

+ Giao tiếp rõ ràng với người dùng bằng hộp thoại messagebox.

+ Phản hồi thời gian thực.

+ Trạng thái hunger và boredom được cập nhật ngay sau mỗi thao tác.

- Hạn chế:

+ Không lưu trạng thái giữa các phiên làm việc.

+ Tính năng đơn giản: Critter chỉ có 2 trạng thái: hunger và boredom. Chưa có biểu hiện cảm xúc, hình ảnh hoặc tương tác phong phú hơn.

+ Giao diện chưa tối ưu cho nhiều màn hình.

**4.2 Kết luận**

Qua quá trình nghiên cứu và triển khai, đề tài “Critter Caretaker với giao diện đồ họa (GUI)” đã hoàn thành đầy đủ các yêu cầu đặt ra:

- Thiết kế và cài đặt thành công một ứng dụng chăm sóc thú ảo đơn giản bằng ngôn ngữ Python, tích hợp với thư viện Tkinter để xây dựng giao diện trực quan.

- Ứng dụng được tổ chức theo mô hình lập trình hướng đối tượng (OOP), đảm bảo tính rõ ràng, linh hoạt và dễ mở rộng.

- Cung cấp các chức năng cơ bản cho người dùng như: tạo thú cưng theo tên, cho ăn, chơi, ngủ, đồng thời cập nhật trạng thái thú cưng theo thời gian thực (hunger, boredom).

- Xử lý hiệu quả các tình huống ngoại lệ như: không nhập tên thú cưng hoặc thao tác khi chưa tạo thú cưng.

Nhìn chung, sản phẩm đã đáp ứng tốt yêu cầu của bài toán, đảm bảo tính tương tác, tính trực quan và mức độ ổn định khi vận hành.

Thông qua việc thực hiện đề tài, em đã củng cố và nâng cao được nhiều kiến thức, cụ thể là:

- Nắm vững cách thiết kế và triển khai lớp (class) trong Python, vận dụng hiệu quả nguyên lý của lập trình hướng đối tượng để tổ chức chương trình một cách khoa học.

- Làm quen với quy trình xây dựng giao diện đồ họa bằng Tkinter, hiểu rõ hơn về cơ chế điều khiển sự kiện (event-driven programming).

- Phát triển tư duy lập trình theo hướng module, có sự phân tách rõ ràng giữa logic xử lý và giao diện người dùng.

- Cải thiện kỹ năng xử lý lỗi, kiểm soát đầu vào và phản hồi hợp lý trong môi trường tương tác.

Mặc dù chương trình đã hoạt động ổn định và hoàn thành đúng mục tiêu ban đầu, nhưng vẫn còn nhiều tiềm năng để mở rộng và nâng cấp. Một số hướng phát triển có thể thực hiện trong tương lai bao gồm:

- Bổ sung các trạng thái cảm xúc của thú cưng (như vui vẻ, buồn bã, mệt mỏi...) để tăng tính chân thực và hấp dẫn.

- Tích hợp hình ảnh, âm thanh hoặc biểu tượng động, giúp giao diện sinh động và thân thiện hơn với người dùng.

- Cho phép lưu và tải trạng thái thú cưng, từ đó duy trì quá trình chăm sóc lâu dài thay vì chỉ trong một phiên làm việc.

- Xây dựng hệ thống nhiều thú cưng, cho phép người dùng quản lý và chăm sóc nhiều đối tượng cùng lúc.

- Triển khai chương trình trên các nền tảng di động hoặc web, mở rộng phạm vi ứng dụng thực tế.

*Tóm tắt chương 4*

*Mô tả quá trình chạy thử chương trình, kiểm tra tính năng: tạo thú cưng, cho ăn, chơi, ngủ và xử lý lỗi người dùng. Ghi nhận các kết quả thực tế và hình ảnh mô phỏng. Phần kết luận tổng hợp những kết quả đạt được, các kỹ năng đã học được và đề xuất hướng cải tiến ứng dụng trong tương lai.*

**TÀI LIỆU THAM KHẢO**

1. Michael Dawson, Python Programming for the Absolute Beginner, 3rd Edition, Course Technology PTR, 2010. (Chapter 8 – Critter Caretaker).
2. Nguyễn Văn Hiếu, Lập trình Python cơ bản đến nâng cao, Nhà xuất bản Khoa học & Kỹ thuật, 2021.